PROSE: Parallel Real-Time Operating System for Secure Environments *

Harish Nag,
T. Mack Stallcup,
Stephen Wheat

Intel Corporation
Enterprise Server Group
Beaverton, OR 97007

Roberta Gotfried,
Glenn Ladd

Hughes Aircraft Company
Rada/Communications
Segment
El Segundo, CA 90245

David Greenberg,
Lance Shuler
David van Dresser

SandiaNational
Laboratories
Albuquerque, NM
87185

Chulsoo Kim

Sunsoft
2550Garcia
Ave., Mountain View, CA
94043-1100

Barney Maccabe

University of
New Mexico
Albuquerque,
NM 87131

Abstract

This paper presents the design considerations of a project to develop a real-time, secure operating system for parallel, heterogeneous processing systems. The system will support software technologies required to develop and effectively use secure scalable and distributed high performance computing technologies. Examples of target technologies include aircraft avionics systems, process control systems, and space-based defense systems.

1 Introduction

Intel, Hughes Aircraft Company and Sandia National Labs are developing a hard real-time, secure operating system for scalable parallel processing systems with heterogeneous node types. This operating system will support embedded processing systems applications and assure multilevel security generally equivalent to the cross section of the Trusted Computer System Evaluation Criteria [5] B2 and B3 level certification.

The operating system is designed to deliver high performance for the support of deterministic, real-time applications on message passing architectures. Fault-tolerance and multiple levels of security will also be supported.

The initial target implementation for this work is the TFLOPS technology-based platforms, however the design will be portable to other Massively Parallel Processor (MPP) architectures.

*This work is supported by DARPA under the following grant numbers: Intel Corporation - Order Nos.: 8120, 8120/2, 8120/4 8247/1; Hughes Aircraft Company - Contract No. N00039-95-C-0016, DARPA Order No. B867; Sandia National Labs - DARPA Order No. C974.

2 Overview of Processing System

PROSE (Parallel Real-Time Operating System for Secure Environments) is targeted toward multi-program processing systems to support a complex military system in ”theater-of-war” situations [3]. Other applications for the system include multi-sensor integrated processors for tactical aircraft and the guidance systems for cruise missiles. Another goal for the project is for the system to be usable in commercial applications such as process control systems.

The systems hardware on a typical platform will include a dependable, non-blocking network that supports timeliness guarantees for message delivery. The hardware also will contain a maintenance subsystem which can detect and isolate faulty hardware and allow reconfiguration of applications around faulty components. The system will support capabilities such as multiple sensor input and control with real-time response, dynamic application instantiation and shutdown, and detection and control of denial of service attacks.

Applications used in a multi-sensor integrated processor system include: sensor fusion, adaptive search/track, Synthetic Aperture Radar (SAR), imaging sensors, electronic warfare subsystems and sonar. The processing system must support a combination of these and other applications where control messages can determine the usage of the resources in a highly dynamic environment.

Operator or sensor generated control messages can cause time critical subsystems to execute by preempting other applications. This processing may require applications to be scaled up to use a larger set of processing nodes to achieve higher performance. The applications may also be scaled down to smaller numbers of nodes in case of node failures in the system. A set of ”spare” nodes could be used to reconfigure applications in case of a hardware failure.

The processing system also adheres to security and trust

requirements as dictated by a cross section of the Trusted Computer System Evaluation Criteria B2 and B3 security levels. It is designed so that, when integrated into a complete system, it will be possible to reach these security levels.

Figure 1 shows a typical processing system with a set of applications executing on a scalable, heterogenous, MPP platform. The applications execute on compute nodes while the I/O and special purpose nodes are used for control and data input from sensors, file I/O, operator commands, and network access.

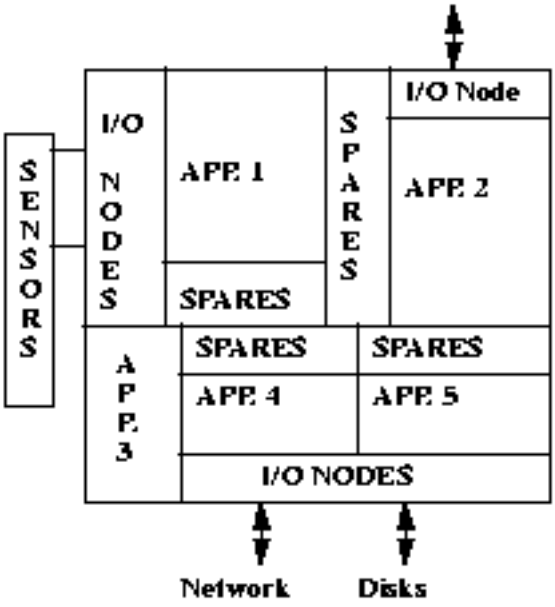

Figure 1. Figure 1.

3 Target Platforms

PROSE is designed for MPP computers. These machines are distributed memory, multi-processor systems consisting of nodes connected by high-speed communication network. Some nodes are used for computing while others are dedicated to I/O or other special purposes. The number of nodes in the system may range from one to several thousand. Each node may contain multiple CPUs and DMA devices which share a common memory system. It is assumed the communication network between the nodes is ”closed”. A closed network in this context means messages on the network can only be observed by the destination node. It also means messages can only be placed on the network by nodes in the system. This network does not include any communications to the outside world such as Ethernet connections. A closed network simplifies access mediation of network messages.

The initial platform for PROSE is a system based on the technology developed by Intel for the TFLOPS machine currently being built for the Department of Energy. The TFLOPS system contains multi-processor nodes connected via a closed network called the Internode Communication Fabric (ICF).

Each node on the TFLOPS system contains two Pentium Pro Processors with a large shared memory. The nodes are connected to the network using an on-board Cavallino Network Interconnect Chip (cNIC). The cNIC contains logic to support DMA transfers between the ICF and memory and vice versa. The ICF consists of Cavallino Mesh Routing Chips (cMRCs) connected in a 2-1/2 dimensional mesh. (A 2-1/2 dimensional mesh consists of two parallel 2-D meshes connected via their respective Z axes.) The messages travelling on the network are routed using Z,X,Y,Z routing parameters. The X,Y, and Z parameters indicate the number of hops in the respective axes from the source node to the destination node.

Each backplane in the system has a Patch Support Board (PSB) which continuously monitors the local resources for any faults. The PSBs are i386-based processing units running a small real-time kernel which interfaces with the local resources.

The PSBs are connected to the Scalable Platform Services (SPS) station. The software on the SPS station communicates with the PSBs to gather monitored information and manage fault handling by detecting and isolating faulty hardware. The SPS station also runs other servers, e.g., diagnostics server, boot server, etc.

This initial platform is not necessarily indicative of the final target platforms for PROSE. Target platforms may be different from the TFLOPS system because of security constraints.

4 PROSE Overview

PROSE is an extension of the Puma technology jointly developed by Sandia National Labs and the University of New Mexico. Puma is the successor to the SUNMOS operating system [4]. Security aspects of PROSE apply technology in real-time embedded systems developed at Hughes Aircraft Company.

The PROSE design is modular with clearly defined functionality and interfaces for each layer. The most privileged layer, the Quintessential Kernel (QK), provides mechanisms to manage the system resources. The Process Control Thread (PCT) manages resources by asking for services from the QK. The library provides interface for user applications to access system services. The layered architecture of PROSE is shown in Figure 2.

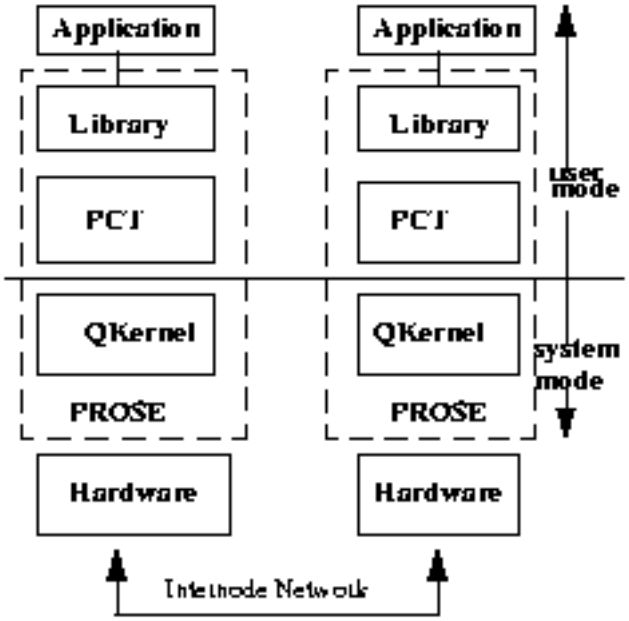

Figure 2. Figure 2.

4.1 Quintessential Kernel

The QK provides mechanisms to enforce the management policies for the system resources but does not actually manage them. It executes in supervisor mode preventing other entities from accessing its memory and data structures.

The QK design was kept small by placing only the functionality which required supervisor privileges into the kernel. The functionalities supported by the QK include: managing virtual address structures, switching between processes, detecting and responding to system events, maintaining message queues and security.

The message queues are used for message passing between processes. The QK validates memory addresses and authenticates message passing requests between different security subjects. Access control tables are used to determine if a subject can access an object, including the sending and receiving of messages. The PROSE design, especially modularity and minimizing the size of QK, also serves to support the security and trust requirements of the system. Design of the QK and PCT ensure that the enforcement mechanisms cannot be bypassed or tampered with.

The hard real-time is achieved by imposing an upper bound on the time required for various system activities, e.g., interrupt (event) handling, process switching and message delivery. High performance is achieved by carefully crafting these sections of the operating system and minimizing transitions between user and supervisor modes.

4.2 Process Control Thread

The PCT is responsible for resource management on the node. It loads processes, allocates memory for them and reclaims node resources upon process completion. It also provides support for application debug, event management and communication management.

The PCT only manages the resources, the mechanisms to control the resources are built in the QK. When the PCT requires a privileged service, it requests the kernel to perform the service. For example, when an application wants to sends a server message (which the PCT must actually send), the PCT verifies the validity of the message and requests the kernel to place the message in the message queue to be sent out. The QK notifies the PCT when the message is sent. This way the two components of the system work together, but the management of resources is separated from the mechanisms.

With this separation of functionality, PROSE achieves a modular design providing for a more secure system. The modularity also provides better availability of the system. Since the QK is persistent, it is always possible to reload a PCT that has faulted. The new PCT can then reload the application, thereby enhancing system availability. This process can take place without resetting a node or reloading the kernel onto the node.

Another advantage is that the PCT is a replaceable part of the system. Since the PCT runs in user space, it is possible to customize the PCT to allow for different resource management schemes, e.g., different process scheduling algorithms. This can occur without changes to the underlying protection mechanisms in the QK.

4.3 Library

PROSE is designed with much of the OS related functionality implemented at the user-level. This allows the library to provide some OS functionality and/or support for other applications to provide this functionality. The functionality supported by libraries includes user-level event handling, management of shared processor interactions, and support for threads.

5 System Requirements

The targeted application systems range from commercial real-time systems with no security requirements to large scale Department of Defense embedded systems supporting multiple applications, large amounts of I/O, and diverse security classifications. These complex systems also require fault tolerance, high availability and maintainability. The following lists the design and development goals for PROSE in priority order.

5.1 Hard Real-Time

Real-time systems are defined as those systems in which the correctness of the system depends not only on the logical result of computation but also on the timeliness of the result. The targeted processing system may support a complex set of applications where the resource allocation is demand driven. The control messages arriving from the sensors or operators can cause preemption of a low priority application in favor of a time-critical, high priority application. In a tactical environment, determinism and the timeliness of message arrival is very important.

Determinism dictates the repeatability of an operation with a high degree of confidence. For example, the system provides support for predictable message arrival on a node by following a pre-determined set of processing steps.

The interrupt (event) handling should have a predetermined upper bound. This allows applications to react to the arriving messages quickly and frees up external channels to receive the next message. It also leads to scheduling other applications which may be time critical.

Preemptive priority scheduling allows system resources to be allocated to applications in a timely manner in response to external stimuli. Support for preemptive priority scheduling, fast interrupt handling and fast delivery of messages allows PROSE to meet hard real-time requirements.

5.2 Security

The concept of computer security involves three aspects [2] : (1) a security policy stating the laws and rules to protect secure information, (2) internal mechanisms to enforce that security policy, and (3) assurance that the mechanisms enforce the security policy.

These aspects pertain to the abstractions used in designing secure computer systems. One abstraction is that of the reference monitor. A reference monitor provides a set of principles which can be used to select security features. These principles guide the implementation so the system provides a high degree of resistance to malicious software.

The reference monitor refers to an abstract machine that mediates accesses by active entities called subjects to passive entities called objects, based on the system security policy. The reference monitor concept is independent of the specific rules that make up the access control policy.

In an embedded system such as PROSE, subjects correspond to processes executing in a particular domain in a computer system. A domain of a process is defined to be the set of objects the process currently has the right to access based on the current access modes.

The security policy deals with controlling subjects including human operator. Mandatory access control (MAC) policies regulate whether and how a subject can access an object. The enforcement of these policies assures a high degree of protection. The MAC policies can provide protection against unauthorized modification of information (integrity) as well as protection against unauthorized disclosure (confidentiality).

Discretionary access control (DAC) policies allow the subjects in a system to specify, at their own discretion, which other subjects in the system shall have access to specific information controlled by the subject. In a system which incorporates both mandatory and discretionary access control policies, the discretionary access control policies serve to provide a finer granularity within the mandatory access control policies.

The Trusted Computing Base (TCB) is defined to be the totality of protection mechanisms within a computer system that are responsible for enforcing the security policy. The security kernel is the most privileged part of the TCB, and it implements the reference monitor.

The security kernel is an instance of the reference monitor and it is a small fraction of the larger operating system software. The operating system should be modular in order to separate the security-related software into a trusted kernel. This allows the remainder of the operating system to rely on the security kernel to ensure the security of the system.

In order to achieve confidence that the kernel ensures the security of the operating system, it must be suitably protected (tamperproof), non-bypassable, and correctly implemented (analyzable.) This last feature means that the size of the kernel must be kept as small as possible and contributes to the system characteristic of trustworthiness. These requirement must be enforced during the design and implementation phases of the system.

While real-time performance is critical to success of the system mission, system reliability and success can be compromised in the presence of security vulnerabilities. Tradeoffs between performance, usability and security continue throughout the development phase of the target system. The modularity of PROSE will allow various trade-off points to be explored and versions to be tailored to each target system.

Some of the security requirements for PROSE include: the support of subjects (applications) and objects at different security levels, erasure of classified data at shutdown and generating and protecting audit records for security-relevant events. It will also provide mandatory access controls over system resources such as memory, I/O devices and buffers, system queues and message queues. In addition, PROSE will support object reuse, identification and authentication of subjects, and will guard against denial of service attacks.

5.2.1 System Build

PROSE is part of the Trusted Computing Base. The QK implements the reference monitor by enforcing the security policies. One essential requirement of the Trusted Computing Base is to minimize the act of mediation or decision-making and still apply the required enforcement during runrime. PROSE achieves this by using a system build approach [1].

The system build approach provides a high level of assurance. It is based on the inherent properties of embedded, real-time systems. These properties include: access relations between security subjects and objects are fully defined before the system is operational and the system does not assume new functionality while in operation. Additional properties inherent in real-time systems include: system applications are limited in number and have well defined functions and most security subjects are application programs performing dedicated tasks, not interfacing with human users.

The system build approach involves performing access mediation at build and initialization time, rather than at run time. Approved accesses are encoded in Privilege Control Table (PCTbl) that is the basis for run-time privilege enforcement by the QK. Because access requests are dealt with before the application runs, run-time penalties due to security functions are minimized. This enables the system to provide security without suffering undue performance penalties.

5.2.2 PROSE Layered Design

PROSE also achieves high level of assurance through the use of a layered architecture that implements the principle of least privilege, e.g., the separation of the QK and the PCT. The layered design of PROSE implements the principle that the code that manages the security objects does not have to be aware of the security attributes of the objects. This helps to separate protection-critical parts of the TCB from non-protection-critical parts.

5.2.3 Security Functionality

PROSE security functionality includes: security labels, including message labels, implementation of MAC, authentication of identification and authorization before beginning to perform any action requiring mediation, and control of resource usage by implementing resource quotas to protect against denial-of-service attacks.

5.3 Scalability

PROSE supports OS functionality for parallel processing systems in which the applications can run on number of nodes varying from one to several thousand. The applications can be reconfigured dynamically to run on a larger number of nodes for time-critical processing or run on a reduced number of nodes due to loss of resources through hardware failure.

5.4 Performance

The target applications of PROSE must achieve high performance subject to the requirement for hard real-time capability and support for multilevel security. In prioritizing capabilities, hard real-time and multilevel security features take precedence over performance in many systems. However, for larger acceptance of PROSE, it is also desirable to give priority to high performance or to even remove all security considerations. For instance, there are no security requirements for the control system in a cruise missile. This flexibility is also a design goal for PROSE.

5.5 Portability

The initial design of PROSE is targeted for an Intel platform based on the TFLOPS technology. However, it should be portable to other MPP architectures. PROSE, and the underlying Puma technology, have been designed to be portable. If an architecture meets the base requirements outlined in [3], then PROSE can be ported to that architectures. The use of message passing scheme makes it easier to port PROSE to other distributed memory systems.

5.6 Fault Tolerance

In order to maintain high system availability, PROSE needs to support fault tolerance. PROSE will support a range of fault tolerance techniques ranging from built-in hardware tests to application level fault tolerance. In some cases, it may not provide these services directly, but instead will provide an infrastructure so fault tolerance capabilities can be built on top of existing operating system functionality.

PROSE terminates faulty applications and if the fault is a security event, it is logged where it can be inspected later. PROSE needs hardware support to be able to reroute messages in case of the hardware failure in the interconnect network. It will also support recovery from single points of failure in the hardware.

5.7 Testability and Maintainability

Support for software and hardware debug and instrumentation is needed for initial development, system integration and maintenance during the lifetime of the system. The system integration phase of the development process requires

debug capability far beyond that which is acceptable during normal operations of the system. PROSE will support the ability to tailor the debug environment to the specific requirements at each phase of operation of the system.

PROSE will also support the maintenance tasks required of tactical systems. This includes the ability to download audit logs and the ability to upload applications and data in conformance with system security policy.

6 Summary

PROSE is designed to support processing systems that can run applications ranging from simple commercial processing to complex military processing.

The teams from Hughes, Intel and Sandia National Labs have completed the Software Requirements Document for the PROSE project. They are currently in the process of implementing PROSE.

References

[1] J. P. Alstad et al. The role of system build in trusted embedded systems. In *Proceedings of the 13th National Computer Security Conference*, volume 1, October 1990.

[2] M. Gasser. *Building a Secure Computer System*. Van Nostrand Reinhold, 1988.

[3] S. N. L. Hughes, Intel. *Software Requirements Specification for the Parallel Real-Time Operating System for Secure Environments (PROSE)*, April 1996. Submitted to DARPA.

[4] A. B. Maccabe et al. Sunmos for the intel paragon: A brief user's guide. In *Intel Supercomputing '94 Proceedings*, pages 245–251, June 1994.

[5] D. of Defense. *DoD 5200.28-STD, Department of Defense Trusted Computer System Evaluation Criteria (TCSEC)*.